Ques: Given an array, and a number k, k < size of array.

→ Find the k^{th} smallest element in the given array.

Eg: arr[] → {7, 10, 4, 3, 20, 13, 6}
k = 3

3 4 6 7 10 13 20

ans = 6

Eg: arr[] = {8, 9, 13, 22, 34, 6}
k = 5
ans = 22

Approach ①

① Sort the array and pick the k^{th} smallest ele.

TC → (n log n)

Approach ②:

a [ ]

Pick a random element (x) and split the array into 2 parts.

x = a[random(0, ..., n-1)]

[ ] < x   [ ] ≥ x

If (k) < size(left partition)
(3)   (6)

↓ first 6 smallest ele exist here.

```
find (l, r, k)
    If (r - l == 1):
        return a(k);
    x = a[random(l, ..., r-1)]
    m = l
    for i = l ... r-1
        If a[i] < x
            swap (a[i], a[m])
            m++;
        If (k < m)
            return find (l, m, k);
        else
            return find (m, r, k);
```

1 recursive call

$\left(n + \frac{2}{3}n + \frac{4}{9}n + \cdots \right) = 3n$

O(n)

Blum - Floyd - Pratt - Rivest - Tarjan algorithm

a [ ]

5 parts

Counting Sort

counts the occurances of each unique ele of the array.

arr → | 4 | 2 | 2 | 8 | 3 | 3 | 1 | 3 → 2

① Find the max ele of array.
max → 8

② Make an array (max + 1) → Initialise all 0 of max

count → | 0 | 1 | 2 | 2 | 1 | 0 | 0 | 0 | 1 |
0 1 2 3 4 5 6 7 8

③ Store cumm sum of the ele of count array.

count → | 0 | 1 | 3 | 5 | 6 | 6 | 6 | 6 | 7 |
0 1 2 3 4 5 6 7 8

0 — 4   0 — 3   0 — 8

1 2 2 3 3 4 8

④ find index of each ele in the orig array after sorting.

array → 4 2 2 8 3 3 1   7 − 1 = 6

count → | 0 | 1 | 1 | 5 | 5 | 6 | | 6 | 6 |
0 1 2 3 4 5 6 7 8

6 − 1 = 5   3 − 1 = 2   2 − 1 = 1

output → | 1 | 2 | 2 | 3 | 3 | 4 | 8 |
0 1 2 3 4 5 6

Time Complexity

Worst case → O(n + k) ≈ O(n)
Best case → O(n + k) ≈ O(n)
Average case → O(n + k) ≈ O(n)

k = max ele

Space Complexity

SC → O(max)

Applications

① Smaller integers with multiple occurances
② linear complexity

| 0 | 1 | 2 | 2 | 1 | 0 | 0 | 0 | 1 |

C

arraylist

| 1 | 2 | 2' | 3 | 3 | 4 | 8 |

Stability

Bucket Sort

→ divides elements into buckets.

→ scatter & gather approach

① The elements are first scattered into buckets
② Then the elements of bucket are sorted
③ elements are gathered in order.

| 11 | 9 | 21 | 8 | 17 | 19 | 13 | 1 | 24 | 12 |

↓ scatter ①

[1] 0 − 5   [8 9] 5 − 10   [11 12 13] 10 − 15   [17 19] 15 − 20   [21 24] 21 − 25

② sort

| 1 | 8 | 9 | 11 | 12 | 13 | 17 | 19 | 21 | 24 |

① 0.42 0.32 0.23 0.52 0.25 0.47 0.51

0.42 * 10 = 4.2 ≈ 4

Create an array of size 10. Sorting algo

| 0 | 0 | 0.23, 0.25 | 0.32 | 0.42, 0.47 | 0.52, 0.51 | 0 | 0 | 0 | |
0 1 2 3 4 5 6 7 8 9

[0 − 1)   0.51; 0.52   quick sort

gathering

| 0.23 | 0.25 | 0.32 | 0.42 | 0.47 | 0.51 | 0.52 |

Time Complexity

Worst case → $O(n^2)$
[0.4 → 0.5] → same bucket

Best case → O(n + k)
10 elements → 10 buckets
1 bucket → 1 element

Average → O(n)

Applications

→ input is uniformly distributed over the range
→ there are floating values